

PROGETTO BASI DI DATI

ManagePizza

**Moretto Nicholas
Seganfreddo Raul**

A.a. 2020/2021

1. Abstract

ManagePizza è un database per la gestione di una pizzeria. Il sistema deve gestire le ordinazioni da parte dei clienti. Il sistema prevede l'acquisto delle pizze per asporto tramite il sito internet: per questo il cliente deve fornire username e password, oltre che il nome. La pizzeria fa servizio anche al tavolo, quindi nella base di dati saranno registrate anche le bibite, e sarà applicato il costo del coperto sul totale.

Oltre al classico menù bisogna gestire l'eventuale aggiunta o rimozione degli ingredienti, e quindi il relativo costo totale della pizza. Inoltre, è necessario memorizzare i singoli ingredienti con la relativa quantità, così da calcolarne il prezzo e decrementare la quantità ogni volta che viene utilizzato, in modo da essere avvisati quando le scorte del magazzino stanno per terminare, e quindi contattare un fornitore.

Il database gestisce anche il conto del locale, che riguarda gli incassi di ogni serata e le varie spese degli stipendi del personale e dei prodotti.

2. Analisi dei requisiti

2.1. Descrizione dei requisiti

Il progetto vuole rappresentare un database sulla gestione delle comande, dei prodotti e dei pagamenti.

La comanda viene suddivisa in ordine_pizza e ordine_bibita. Ogni pizza ha il suo nome e sarà composta da alcuni ingredienti che verranno specificati, in modo da calcolarne il prezzo anche in caso di aggiunte o rimozioni di ingredienti. Per le bibite, invece, sarà presente il nome e il relativo prezzo.

Il cliente nel caso in cui consuma al tavolo, non è una persona fisica, ma l'insieme di persone che si siede al tavolo. Per risalire quindi alle ordinazioni di un gruppo di persone in una serata basta risalire al loro id_cliente.

Il cliente che effettua l'ordine si divide in cliente d'asporto e cliente in sala. Il cliente in sala è caratterizzato dal numero del tavolo e sarà servito da un cameriere specifico, il quale potrà accettare mance nel caso gli venissero offerte. Il cliente per asporto, invece, dovendo ordinare online, dovrà registrarsi nel sito e fornire:

- Username
- Password
- Nome
- Ordine delle pizze

Al cliente che ordina online gli verrà fornito il conto che dovrà pagare nel momento in cui va a prendere le pizze in pizzeria, e l'orario in cui le pizze dovrebbero essere pronte.

L'ordine elabora uno scontrino, in cui viene indicato il numero dello scontrino, la data, il prezzo delle pizze e quello delle bibite. Però lo scontrino per i clienti in sala sarà leggermente diverso da quello per i clienti d'asporto. Infatti, ai clienti in sala verrà aggiunto il coperto.

Per le merci quali ingredienti e bibite, verrà tenuto conto della quantità presente in totale, quindi anche nel magazzino, e nel caso in cui questa quantità dovesse andare sotto un certo numero, apparirà un messaggio che invita il titolare a chiamare al più presto un fornitore.

Ogni fornitore sarà caratterizzato dal numero di Partita Iva, ragione sociale e indirizzo.

Gli ordini dei clienti verranno incassati dal locale, il quale avrà il conto con cui effettuerà tutti i pagamenti. Nei pagamenti sono presenti le spese dei prodotti e gli stipendi del personale.

2.2. Glossario dei Termini

Termine	Descrizione	Collegamento
Cliente	Persone che effettuano le comande alla pizzeria	Collegata ad Asporto, Tavoli e comanda
Asporto	Clients che effettuano le comande per consumare i prodotti altrove	Collegata a Cliente
Tavoli	Clients che effettuano le comande per consumare i prodotti all'interno della pizzeria	Collegata a Cliente
Comanda	Ordinazione dei prodotti da parte dei clienti	Collegata a Cliente, Scontrino, Bibite, Pizze
Scontrino	Ricevuta fiscale con il riepilogo e prezzo della comanda effettuata	Collegata a Comanda e Conto
Bibite	Bevande disponibili per l'acquisto	Collegata a Comanda e Magazzino
Pizze	Pizze presenti nel listino	Collegata a comanda e Pizze_ingredienti
Pizze_ingredienti	Lista delle pizze con relativi ingredienti	Collegata a Pizze e Magazzino
Magazzino	Insieme di ingredienti e bibite già acquistati	Collegato a Ordini, Pizze_ingredienti e Bibite
Fornitore	Commerciante che vende ingredienti e bibite alla pizzeria	Collegato a Ordini e Fatture
Fatture	Documento fiscale che attesta l'acquisto delle merci dai fornitori	Collegato a Fornitore e Conto
Ordini	Lista delle merci da acquistare dai fornitori	Collegato a Fornitore e Magazzino
Conto	Conto bancario della pizzeria: qui sono presenti tutte le entrate e le uscite	Collegato a Fatture, Scontrino e Personale
Personale	Sono i dipendenti della pizzeria: dal titolare ai pizzaioli ai camerieri	Collegato a Conto

3. Progettazione concettuale

3.1 Lista delle entità

Il database rappresenta le seguenti classi. L'identificatore delle entità viene sottolineato e viene sott'inteso che sia *not null*.

- **Cliente:** rappresenta le persone che effettuano le ordinazioni
 - id_cliente: serial primary key
 - tipo: boolean (True=al tavolo, False=d'asporto) not null
- **Asporto:** rappresenta le ordinazioni per i clienti che consumano i prodotti al di fuori della pizzeria

- utente: varchar(16) primary key
- password: varchar(16) not null
- nome: varchar(16) not null
- id_cliente: int primary key
- **Tavolo**: rappresenta le ordinazioni per i clienti che consumano i prodotti in pizzeria
 - n_tavolo: int primary key
 - n_coperti: int not null
 - id_cliente: int not null
 - id_cameriere: int not null
- **Comanda**: rappresenta le ordinazioni dei clienti
 - id_comanda: serial primary key
 - nome_pizza: varchar(32)
 - nome_bibita: varchar(32)
 - id_cliente: int not null
 - quantita: int not null
- **Scontrino**: rappresenta la ricevuta fiscale
 - n_scontrino: int primary key
 - id_cliente: int not null unique
 - data: timestamp not null
 - prezzo_pizze: decimal(6,2)
 - prezzo_bibite: decimal(6,2)
- **Bibite**: rappresenta tutte le bevande destinate alla vendita
 - nome: varchar(32) primary key
 - importo: decimal (4,2) not null
- **Pizze**: rappresenta il listino delle pizze
 - nome: varchar(32) primary key
 - importo: decimal(4,2) not null
- **Pizze_ingredienti**: lista delle pizze con i relativi ingredienti
 - nome: varchar(32) primary key
 - ingrediente: varchar(32) primary key
- **Magazzino**: inventario degli ingredienti e bibite a disposizione per essere utilizzati
 - prodotto: varchar(32) primary key
 - bibita_o_ingrediente: boolean (T=bibita, F=ingrediente) not null
 - quantita: int not null
 - scorta_min: int not null
 - importo: decimal(4,2) not null
- **Ordini**: rappresenta gli ordini che vengono fatti ai fornitori per colmare il magazzino
 - id_ordine: serial primary key
 - fornitore: char(11) not null
 - quantita: int not null
 - prodotto: varchar(32) not null
 - bibita_o_ingrediente: boolean (T=bibita, F=ingrediente) not null
 - data: timestamp not null
- **Fornitore**: azienda a cui vengono effettuati gli ordini
 - p_iva: char(11) primary key
 - ragione_sociale: varchar(32)
 - indirizzo: varchar (32)
- **Fatture**: documento fiscale emesso dal fornitore
 - n_fattura: int primary key
 - bibita o ingrediente: boolean (T=bibita, F=ingrediente) not null
 - fornitore: char(11) not null

- data: timestamp not null
- importo: decimal(6,2) no null
- **Conto:** rappresenta il conto bancario della pizzeria, in cui sono presenti tutti i movimenti
 - id_operazione: serial primary key
 - tipo: boolean (T=entrata, F=uscita) not null
 - data: timestamp not null
 - n_fattura: int
 - n_scontrino: int
 - importo: decimal (6,2) not null
 - salario: int
- **Personale:** dipendenti della pizzeria
 - id_personale: serial primary key
 - nome: varchar(16) not null
 - cognome: varchar(16) not null
 - stipendio: decimal (7,2) not null
 - mancia: decimal (4,2)

3.2 Tabella delle relazioni

Relazione	Entità coinvolte	Descrizione	Attributi
effettua	Cliente (1,N) Comanda (1,1)	Un cliente può effettuare più comande e una comanda può essere effettuata da un solo cliente.	nessun attributo
elabora	Comanda (1,1) Scontrino (1,N)	Una comanda può elaborare un solo scontrino. Uno scontrino può essere elaborato da più comande.	nessun attributo
è composta da	Comanda (1,N) Pizze (1,N) Bibite (1,N)	Una comanda può essere composta da più pizze e da più bibite, e una pizza o una bibita può essere presente in più comande.	nessun attributo
composte da	Pizze (1,N) Pizze_ingredienti (1,1)	Una pizza può essere composta da più ingredienti di quella pizza, e un ingrediente di quella pizza fa parte solo di quella pizza.	nessun attributo
contenuti in	Pizze_ingredienti (1,1) Magazzino (1,N)	Un ingrediente è contenuto nel magazzino, e il magazzino può contenere più ingredienti.	nessun attributo
richiede	Magazzino (1,N) Ordini (1,1)	Il magazzino può richiedere più ordini, e un ordine è richiesto da un solo magazzino.	nessun attributo
effettuati a	Ordini (1,1) Fornitore (1,1)	Un ordine può essere effettuato a un solo fornitore, e un fornitore può effettuare un solo ordine.	nessun attributo
emette	Fornitore (1,N) Fatture (1,1)	Il fornitore può emettere più fatture, e una fattura può essere emessa da un solo fornitore.	nessun attributo
addebito	Fatture (1,1) Conto (1,N)	Una fattura viene addebitata sul conto, e nel conto possono essere addebitate più fatture.	nessun attributo
salario	Conto (1,N) Personale (1,1)	A un dipendente viene pagato il salario tramite il conto, e nel conto vengono pagati tutti i dipendenti.	nessun attributo

accredita	Scontrino (1,1) Conto (1,N)	Uno scontrino viene accreditato sul conto, e sul conto vengono accreditati più scontrini.	nessun attributo
------------------	--------------------------------	---	------------------

3.3 Generalizzazioni

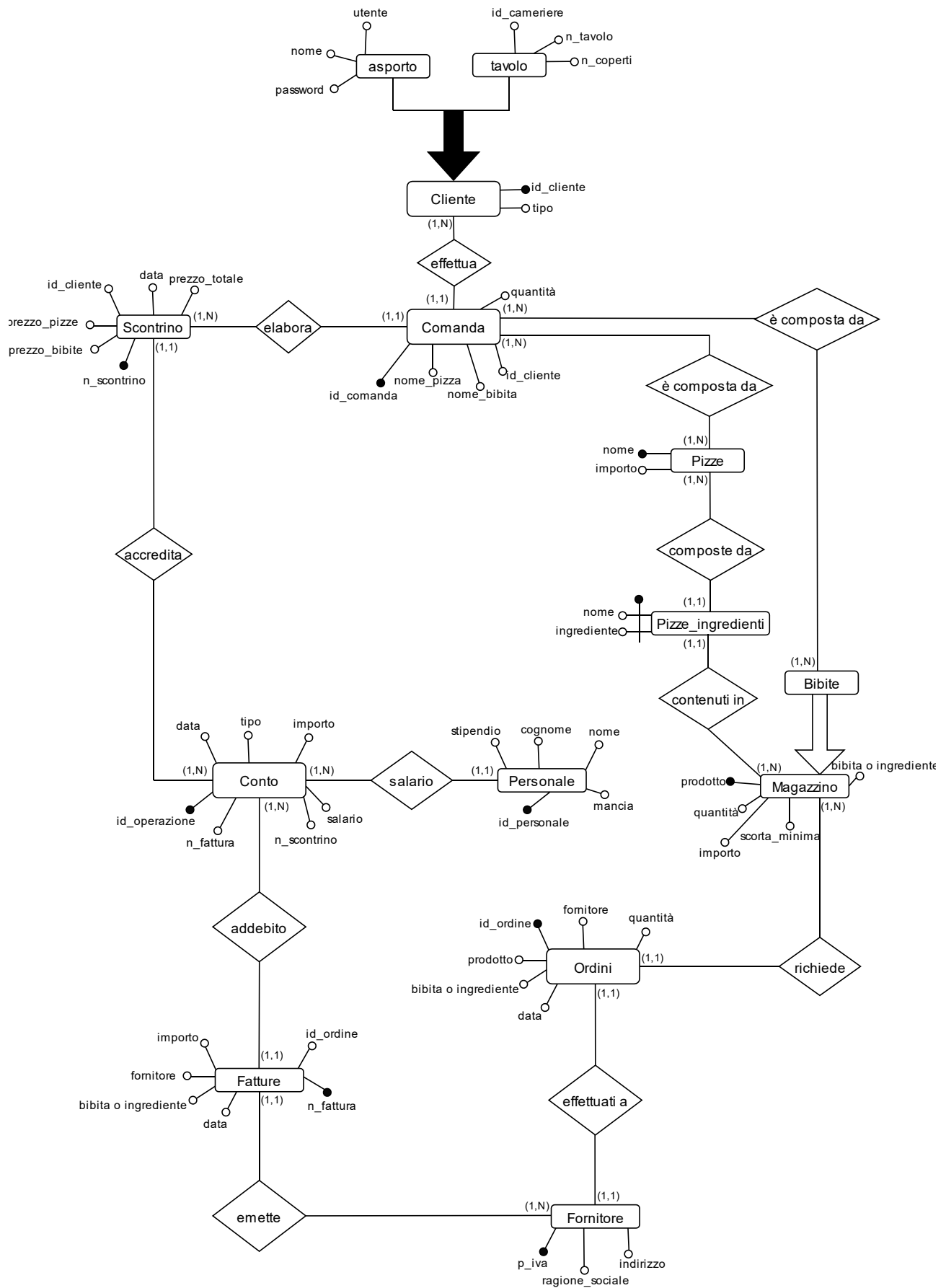
Cliente \leftarrow Asporto
 Cliente \leftarrow Tavolo

Un cliente può ordinare da asporto o direttamente in sala: per questo viene suddiviso in due sottocategorie con una generalizzazione totale.

Magazzino \leftarrow Bibite

Nel magazzino sono rappresentate le bibite con il loro importo: per questo possiamo considerare le bibite come un loro sottoinsieme.

3.4 Schema concettuale (E-R)



4. Progettazione logica

4.1 Ristrutturazione dello schema







4.1.1 Analisi delle ridondanze

Il campo `prezzo_totale` dell'entità `scontrino` crea una ridondanza in quanto questo dato può essere ricavato sommando il totale dei prezzi delle pizze e delle bibite, campi già presenti in `scontrino`, e aggiungendo il prezzo del coperto, dato che si ricava moltiplicando il numero dei coperti presenti nella tabella `tavolo` (`n_coperti`) per il prezzo di un coperto.

Query per ottenere il totale di ogni `scontrino`:

```
select n_scontrino, s.id_cliente, prezzo_pizza, prezzo_bibite, n_coperti,  
(prezzo_pizza + COALESCE(prezzo_bibite,0) + COALESCE(n_coperti*1.5,0)) as totale  
from Scontrino as s full join Tavolo as tav on s.id_cliente = tav.id_cliente
```

L'output ottenuto è il seguente:

Data Output		Explain	Messages	Notifications		
 n_scontrino integer	 id_cliente integer	 prezzo_pizza numeric (6,2)	 prezzo_bibite numeric (6,2)	 n_coperti integer	 totale numeric	
1	5	1	27.50	16.30	5	51.30
2	4	2	22.00	10.50	4	38.50
3	7	5	60.00	35.00	8	107.00
4	6	6	21.00	8.50	3	34.00
5	3	7	25.50	[null]	[null]	25.50
6	2	4	24.00	[null]	[null]	24.00
7	1	3	26.00	[null]	[null]	26.00

4.1.2 Eliminazione delle generalizzazioni

Generalizzazione	Risoluzione
Cliente ← Asporto, Tavolo	Si sceglie di creare due tabelle separate in quanto sono entità distinte (Asporto, Tavolo), nel senso che avranno due trattamenti diversi nella gestione delle comande e nel pagamento finale.
Magazzino ⇌ Bibite	La tabella bibite è necessaria per effettuare le comande, quindi verrà creata una relazione con Magazzino.

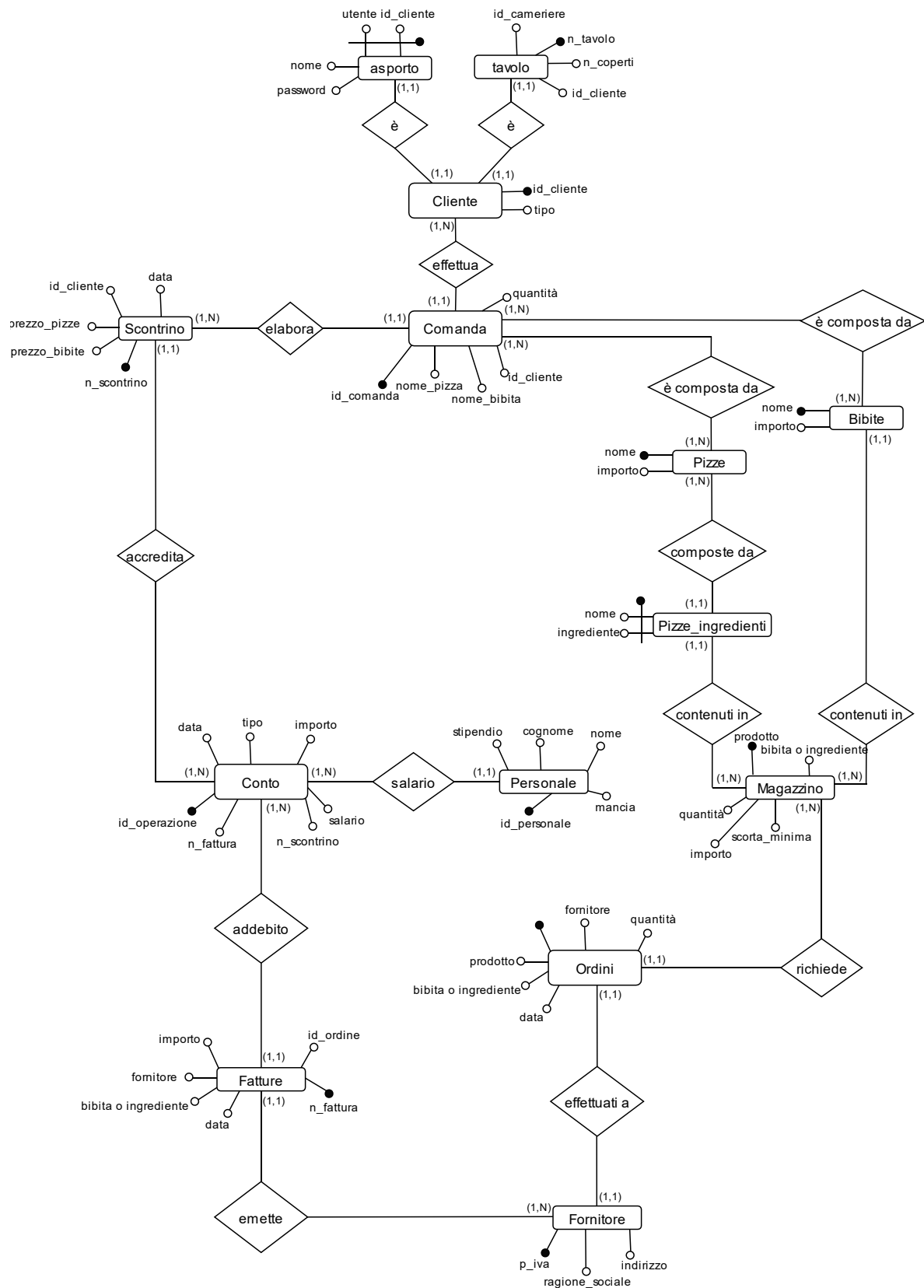
4.1.3 Scelta degli identificatori primari

Una chiave primaria non scontata è la chiave primaria della tabella `pizze_ingredienti`. Si è optato per la combinazione del campo `nome` e del campo `ingredienti`, per avere la possibilità di avere un numero qualunque di ingredienti per ogni pizza e per sapere le quantità di ingredienti che vengono consumati. Se avessimo scelto di tenere come chiave primaria solo il campo `nome` avremmo avuto un certo numero di colonne di ingredienti, e, oltre ad avere un numero massimo fisso di ingredienti possibili, si avrebbe avuto difficoltà a tenere conto delle

quantità.

Inoltre, anche nella tabella asporto si è scelta come chiave primaria l'unione dei campi id_cliente e utente: questo perché l'ordinazione da asporto viene effettuata online, e quindi oltre al fatto che l'id_cliente deve essere univoco, anche l'utente non deve essere ambiguo.

4.1.4 Diagramma schema ristrutturato



4.2 Schema relazionale

cliente (id_cliente, tipo)

asporto (utente, id_cliente, password, nome)

tavolo (n_tavolo, n_coperti, id_cliente, id_cameriere)

magazzino (prodotto, bibita_o_ingrediente, quantita, scorta_min, importo)

pizze (nome, importo)

bibite (nome, importo)

pizze_ingredienti (nome, ingrediente)

scontrino (n_scontrino, id_cliente, data, prezzo_pizza, prezzo_bibite)

comanda (id_comanda, nome_pizza, nome_bibita, id_cliente, quantita)

composta_da_pizze (comanda_id, nome_pizza)

composta_da_bibite (comanda_id, nome_bibita)

fornitore (p_iva, ragione_sociale, indirizzo)

ordini (id_ordine, fornitore, quantita, prodotto, bibita_o_ingrediente, data)

fatture (n_fattura, bibita_o_ingrediente, fornitore, data, importo, id_ordine)

personale (id_personale, nome, cognome, stipendio, mance)

conto (id_operazione, tipo, data, n_fattura, n_scontrino, importo, salario)

asporto.id_cliente → cliente.id_cliente

tavolo.id_cliente → cliente.id_cliente

bibite.nome → magazzino.prodotto

pizze_ingredienti.nome → pizze.nome

pizze_ingredienti.ingrediente → magazzino.prodotto

comanda.id_cliente → cliente.id_cliente

comanda.id_cliente → scontrino.id_cliente

composta_da_pizze.comanda_id → comanda.id_comanda

composta_da_pizze.nome_pizze → pizze.nome

composta_da_bibite.comanda_id → comanda.id_comanda

composta_da_bibite.nome_bibite → bibite.nome

ordini.fornitore → fornitore.p_iva

ordini.prodotto → magazzino.prodotto

fatture.fornitore → fornitore.p_iva

conto.n_fattura → fatture.n_fattura

conto.n_scontrino → scontrino.n_scontrino

conto.salario → personale.id_personale




fatture.id_ordine → ordine.id_ordine

5. Query e indice

5.1 Query

1. Calcolare i ricavi della pizzeria svolti nel corso della giornata. Si può dare la possibilità al datore di lavoro di scegliere il giorno da visualizzare.




```
select DATE(Scontrino.data) as Date,  
(sum(prezzo_pizza) + sum(prezzo_bibite) + sum(COALESCE(n_coperti*1.50,0))) AS ricavo_totale  
FROM Scontrino FULL JOIN Tavolo ON Scontrino.id_cliente = Tavolo.id_cliente  
WHERE DATE(Scontrino.data) = '2020-12-20'  
group by Date
```

	Data Output	Explain	Messages	Notifications
	 date date		ricavo_totale numeric	
1	2020-12-20		306.30	

Nel caso si volesse scegliere di visualizzare il giorno corrente, si può semplicemente utilizzare la dicitura *current_date* al posto dell'operatore di uguaglianza.

2. Visualizzare le pizze aventi l'ultimo ingrediente ordinato ai fornitori in ordine di tempo.

```
select pi.nome, Ordini.prodotto  
from Pizze_Ingredienti as pi  
full join Magazzino on pi.ingrediente = Magazzino.prodotto  
full join Ordini on Magazzino.prodotto = Ordini.prodotto  
where Ordini.data = (select MAX(Ordini.data) from Ordini)
```

	Data Output	Explain	Messages	Notifications
	 nome character varying (32)		prodotto character varying (32)	
1	4 formaggi		asiago	

3. Nella tabella Conto, dato un importo maggiore o uguale a 200, trovare a chi è indirizzato il pagamento (se a un fornitore per un ordine effettuato o al personale per il pagamento degli stipendi), e visualizzare la ragione sociale del fornitore o il cognome del dipendente a cui è stato effettuato, e la data in cui questo pagamento è avvenuto.

```
select c.importo, forn.ragione_sociale, p.cognome, c.data
from Personale p right join Conto c on p.id_personale = c.salario
full join Fatture on c.n_fattura = Fatture.n_fattura
left join Fornitore forn on Fatture.fornitore = forn.p_iva
where c.importo >= 200
```

Data Output Explain Messages Notifications

	importo numeric (6,2)	ragione_sociale character varying (32)	cognome character varying (16)	data timestamp without time zone
1	200.00	Bevo e non bevo S.p.A.	[null]	2020-12-12 16:31:45
2	200.00	Bevo e non bevo S.p.A.	[null]	2020-12-12 16:38:25
3	300.00	Latterie Venete S.n.c.	[null]	2020-12-13 11:24:45
4	647.45	[null]	Morandin	2021-01-10 09:02:26
5	435.50	[null]	Albino	2021-01-10 09:09:58
6	351.80	[null]	Abate	2021-01-10 09:15:14
7	556.65	[null]	Tigullio	2021-01-10 09:19:32
8	475.90	[null]	Del Vecchio	2021-01-10 09:25:43

4. Trovare il numero di prodotti consumati in un determinato giorno e il numero di prodotti rimanenti in magazzino, ordinati per prodotto in ordine alfabetico.

```
select m.prodotto, SUM(c.quantità) as sum_quantità,
(m.quantità - SUM(c.quantità)) as prodotti_rimanenti
from Scontrino s full join Comanda c on s.id_cliente = c.id_cliente
full outer join Bibite b on c.nome_bibita = b.nome
full outer join Pizze p on c.nome_pizza = p.nome
left join Pizze_Ingredienti pi on p.nome = pi.nome
left join Magazzino m on pi.ingrediente = m.prodotto or b.nome = m.prodotto
where m.prodotto is not null and c.quantità is not null and DATE(s.data) = '2020-12-20'
group by m.prodotto
order by m.prodotto asc
```

	Data Output	Explain	Messages	Notifications
	prodotto [PK] character varying (32)		sum_quantità bigint	prodotti_rimanenti bigint
1	acciughe		3	97
2	acqua frizzante 1.5L		5	145
3	acqua naturale 1.5L		4	196
4	birra bionda 0.3L		5	195
5	birra bionda 0.5L		12	288
6	birra rossa 0.3L		13	187
7	capperi		3	97
8	carciofi		1	149
9	carote		1	149
10	cipolla		2	148

Anche qui se si volesse scegliere di visualizzare la data corrente, basterebbe utilizzare la funzione *current_date*.

5. Data una pizza ordinata (in questo esempio una Rustica), trovare se il cliente che ha ordinato questa pizza è presente in sala o se ha ordinato per asporto. Mostrare a video rispettivamente il numero del tavolo in cui è seduto e il nome del cliente.

```
SELECT c.id_cliente, a.nome, t.n_tavolo
FROM Comanda c JOIN Asporto a ON c.id_cliente = a.id_cliente
LEFT JOIN Tavolo t ON c.id_cliente = t.id_cliente
WHERE c.nome_pizza = 'Rustica'
```

Data Output	Explain	Messages	Notifications
	id_cliente integer	nome character varying (16)	n_tavolo integer
1	3	Ginevra	[null]

6. Trovata la bibita avente quantità minore nel magazzino, stampare a video qual è la fattura relativa all'ordine di quel prodotto, qual era la quantità presente nel magazzino e il fornitore.

```
select ft.n_fattura, m.prodotto, m.quantita, ft.fornitore
from Fatture ft join Ordini o on ft.id_ordine = o.id_ordine
join Magazzino m on o.prodotto = m.prodotto
where m.bibita_o_ingrediente = TRUE and m.quantita = (select MIN(m.quantita)
                                                    from Magazzino m
                                                    where m.bibita_o_ingrediente = TRUE)
```

Data Output	Explain	Messages	Notifications
	n_fattura integer	prodotto character varying (32)	quantita integer
1	7	gingerino	70
			00075625478

Per quanto riguarda il file .cpp, per il corretto funzionamento dello stesso, il Database deve essere creato e popolato attraverso lo script fornito. Dopo di che, bisogna assicurarsi che i “**#define**” siano coerenti con quanto impostato nella propria macchina. Questo perchè per il corretto funzionamento del codice C++, c'è bisogno della libreria “**libpq-fe**” che abilita all'utilizzo di funzioni che permettono di interagire con il Database. Dopo aver fatto tutto ciò, si può compilare lo script ed eseguire il file eseguibile creato per interrogare il database con le query soprastanti.

5.2 Indici

Da non sottovalutare è la ricerca delle fatture, che con il passare del tempo diventeranno sempre di più, in quanto potrebbero sorgere delle problematiche con i fornitori tali che diventa necessario recuperare una specifica fattura. Quindi si predispone che la ricerca sia specifica e veloce.

```
drop index if exists ricerche_su_fatture;
create index ricerche_su_fatture on Fatture(n_fattura, fornitore, importo)
```

Allo stesso modo anche la ricerca degli scontrini potrà necessitare di un indice, in quanto gli scontrini aumenteranno ancora più velocemente e ci potrà essere bisogno di cercare un determinato scontrino, o tutti gli scontrini emessi in un determinato giorno.

```
drop index if exists ricerche_su_scontrini;
create index ricerche_su_scontrini on Scontrino(n_scontrino, prezzo_pizza, prezzo_bibite, data)
```

Con questi indici la ricerca su centinaia di fatture o migliaia di scontrini sarà più rapida e specifica.